REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 1994	Professional Paper

4. TITLE AND SUBTITLE

EVOLVING CASCADE-CORRELATION NETWORKS FOR TIME-SERIES FORECASTING

5. FUNDING NUMBERS

PR: ZW67
PE: 0601152N
WU: DN303002

6. AUTHOR(S)

J. R. McDonnell, D. E. Waagen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Command, Control and Ocean Surveillance Center (NCCOSC)
RDT&E Division
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Chief of Naval Research
Arlington, VA 22217–5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

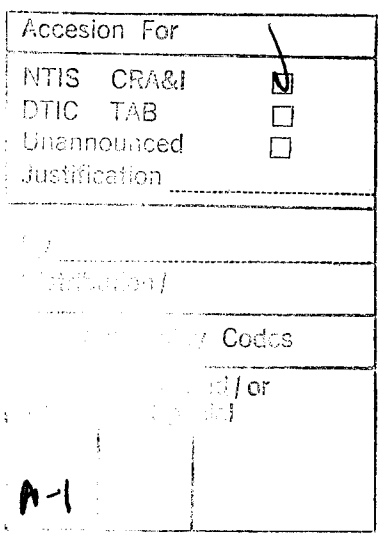
This investigation applies evolutionary search to the cascade-correlation learning network. Evolutionary search is used to find both the input weights and input connectivity of candidate hidden units. A time-series prediction example is used to demonstrate the capabilities of the proposed approach.

Published in *Proceedings of the Third Golden West Int. Conf. on Intelligent Systems*, June 1994.

14. SUBJECT TERMS		15. NUMBER OF PAGES
Evolutionary Search Cascade-Correlation Learning Architecture Time-Series Models	Neural Networks	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE *(include Area Code)*	21c. OFFICE SYMBOL
J. R. McDonnell	(619) 553–5762	Code 786



EVOLVING CASCADE-CORRELATION NETWORKS FOR TIME-SERIES FORECASTING·

J.R. McDONNELL and D.E. WAAGEN†
NCCOSC, RDT&E Div., San Diego, CA 92152
†Systems Integration Group, TRW, Ogden, UT 84403
mcdonn@cod.nosc.mil dwaagen@ow009.bmd.trw.com

Abstract. This investigations applies evolutionary search to the cascade-correlation learning network. Evolutionary search is used to find both the input weights and input connectivity of candidate hidden units. A time-series prediction example is used to demonstrate the capabilities of the proposed approach.

Keywords. Evolutionary Search, Cascade-Correlation Learning Architecture, Time-Series Models, Neural Networks.

1 Introduction

Time-series forecasting is concerned with making future predictions about a process based on its past behavior. Real applications exist in many fields including economics, engineering, and science. Since linear time-series models are not always appropriate when attempting to model an unknown system, a multitude of nonlinear modeling techniques have been proposed (e.g., Casdagli and Eubank, 1992). For example, artificial neural networks are one common class of nonlinear time-series models as discussed by Cichocki and Unbehauen (1994). However, one drawback with applying a purely nonlinear modeling approach to an unknown system is that a linear process may not be easily represented in a nonlinear model (Weigend and Gershenfeld, 1993). The cascade-correlation learning architecture (CCLA) developed by Fahlman and Lebiere (1990) represents a modeling approach which accommodates both linear and nonlinear structure within a single model. The presented work discusses how evolutionary search can be used for generating traditional CCLAs as well as pruned CCLAs. The benefits of pruning the CCLA during construction are realized in a parsimonious model with potentially better generalization capabilities.

The present investigation is concerned only with auto-regressive (AR) type models. A linear AR model for next-step predictions is given by

$$\hat{x}(k) = \sum_{i=1}^{M} a_i x(k-i) + e(k)$$

where M is the order of the AR model, a represents the coefficient associated with each tapped-delay, and e is the forcing function or a white noise term. Evolutionary search has been previously applied by D. Fogel (1991) for determining both the model order and coefficients of linear time-series models. A predictive, nonlinear AR model is described by

---

· *presented at the Third Golden West International Conf. on Intelligent Systems, Las Vegas, NV, 1994.*

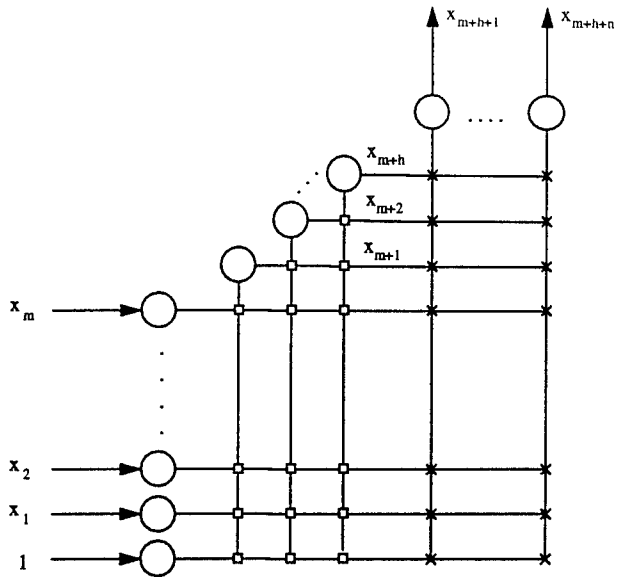


Figure 1. *The cascade-correlation network structure. The boxes □ indicate weights which are frozen when the unit is incorporated into the network. The crosses × indicate weights which are modified after a new hidden unit is incorporated.*

$$\hat{x}(k) = f(x(k-1),...,x(k-M)) + e(k)$$

where *f( )* is the nonlinear mapping. The present investigation is concerned with the combined linear+nonlinear AR model

$$\hat{x}(k) = \sum_{i=1}^{M} a_i x(k-i) + \sum_{j=1}^{N} b_j f_j(x(k-1),...,x(k-M)) + e(k)$$

which may incorporate more than one nonlinear component. This is the essence of the CCLA.

2 The Cascade-Correlation Learning Architecture

The cascade-correlation learning algorithm was introduced by Fahlman and Lebiere (1990) as a means of automatically constructing feedforward networks by adding hidden units until a desired mapping is achieved. The network is initialized with only the input units mapped directly to the output units. New hidden units are added on an individual basis (one-by-one) into the network until a termination criterion is met. Each new hidden unit is selected from a pool of candidate units and is incorporated into the network with its input weights frozen. Using a pool of candidate units increases the likelihood of finding a good hidden unit by reducing the susceptibility to poor initial conditions. The goal of each candidate unit in the population pool is to maximize the correlation between its output and the residual output error of the network. This is the correlation aspect of the cascade-correlation algorithm. Each newly incorporated hidden unit is connected to *all* input

and previously generated hidden units in the network as shown in Figure 1. Likewise, each newly incorporated hidden unit is also fully-connected to all of the output units. After each hidden unit is inserted into the network, training takes place on the output layer weights with the hidden unit weights remaining fixed.

3 Evolutionary search

Evolutionary search algorithms are based on the paradigm of natural evolution as an optimization technique. Evolutionary algorithms (EAs) encompass a variety of multi-agent stochastic search techniques including evolutionary programming (L. Fogel et al., 1966; D. Fogel, 1991; D. Fogel, 1992), evolution strategies (Schwefel, 1981; Bäck and Schwefel, 1993), and genetic algorithms (Goldberg, 1989; Holland, 1992). The presented work employs a modified version of evolutionary programming (EP) by adding a recombination operator to the EP paradigm. The resultant algorithm is very similar to an evolution strategy (ES). A general evolutionary search algorithm is given in Figure 2.

In 1958, Brooks described a *creeping random* method where k points were generated via Gaussian perturbations about a search point. The best point was kept and the process repeated. Brooks (1958) observed that "there are some rather intriguing analogies that can be made between the creeping random method and evolution." This analogy was also apparent to L. Fogel et al. (1966) who applied a random search strategy termed *evolutionary programming* (EP) to the optimization of finite state machines. More recently, D. Fogel (1991; 1992) has extended the EP paradigm to address combinatorial and real-valued function optimization problems as well as applied it to a variety of problems in system identification and control.

```
evolutionary search procedure
begin
k=0
initialize parent population P(k)
evaluate P(k)
do {
        generate offspring O(k) from P(k)
        evaluate O(k)
        select P(k+1) from {P(k) ∪ O(k)}
        k=k+1
} while (terminate condition not met)
end
```

Figure 2. An evolutionary search algorithm.

4 Neural network complexity

Parsimonious networks are less susceptible to overfitting a sample data set and thereby usually result in neural networks with better generalization capabilities. Techniques employed for generating parsimonious structures include increasing network size by the addition of hidden units/layers or pruning an oversized, trained network to yield a smaller network. The CCLA falls into the former class of approaches. Since it is desirable to construct parsimonious architectures, a cost or objective function is formulated which incorporates both system performance and model complexity.

The common form of such a risk function is given by Haykin (1994) as $R(w)=E_s(w)+\lambda E_c(w)$ where E_s is the standard performance measure, E_c is the complexity penalty, and λ is the regularization parameter. The standard performance measure is usually the sum-squared error. As noted by Haykin (1994), there exists a similarity between the risk function $R(w)$ and the composition of statistically derived complexity terms like the minimum description length (MDL) complexity criterion formulated by Rissanen (1986). Initially, $\lambda=0$ so that performance is not sacrificed at the expense of complexity. The regularization parameter λ may be adapted during the training process using a technique described by Weigend et al. (1992).

5 Evolving cascade-correlation architectures

Evolving the weights and connectivity structure of a population of candidate hidden units in a cascade-correlation network is computationally less demanding than evolving the weights and connectivity of a population of neural networks. A valid criticism of this approach is that in exchange for these computational savings, one is solving only small parts of a large problem and thereby reducing the benefits associated with using global search methods on large scale optimization problems. However, the combinatorial optimization capabilities of evolutionary algorithms are beneficial when determining the input weights and connections of the candidate nodes.

A population of candidate hidden units is randomly initialized with full connectivity as in the standard CCLA. The same optimization objective (the absolute value of the covariance between the hidden unit and the residual error) used to train CCLA units is employed to represent the fitness of each candidate node

$$S(a_i) = \sum_o \left| \sum_p (z_{i,p} - \bar{z}_i)(e_{p,o} - \bar{e}_o) \right|$$

where z_p represents the output of each candidate node for pattern p and e is the residual error as measured at the network's output unit o. The weight vector for each candidate unit is modified using standard EP (D. Fogel, 1991)

$$w'_{i,j} = w_{i,j} + \sqrt{Sf / S(a_i)} \cdot N(0,1)$$

where Sf represents the arbitrarily selected scaling factor. Recombination takes place according to

$$\mathbf{w}_k = \mathbf{w}_i + \alpha(\mathbf{w}_j - \mathbf{w}_i)$$

where the indices and scaling coefficient are selected at random such that $i,j \in \{1,..,\mu\}$, $i \neq j$, and $\alpha \sim U(0,1)$. Selection is deterministic so that a new parent set is formed from the best μ members of the set comprised of the original parents, the mutated offspring, and the recombined offspring. After an arbitrary number of generations, the best candidate unit is inserted into the network.

The objective function which incorporates the unit's complexity is formulated using the standard CCLA performance measure $S(a_i)$ in the risk function $R(w)$. Thus the complexity cost retains the general form of the risk function and is given by

$$\Phi(a_i) = -S(a_i) + \lambda E_{c,N}(w,k)$$

where $E_{c,N}(w,k)$ represents the complexity cost for N samples and k parameters. The MDL complexity term $E_{c,N}(w,k)=0.5 \cdot k \log N$ was employed since the connections are strongly specified. The variable complexity regularization parameter λ is tied to the risk Φ of the best candidate node in the population at each generation. No degree of optimality is implied by the given formulation of $\Phi(a_i)$. Although not yet implemented, a reasonable stopping criterion would be to test the current model performance and the newly generated model performance on a validation set. When the current model has superior performance to the newly created model, training is stopped and no more units are added.

All candidate nodes are initially fully connected. Diversity is introduced in the mutation step of the algorithm by randomly selecting an input connection to a candidate node and flipping its bit (i.e., 1→0 or 0→1). Recombination of the connectivity arrays is limited to the bitwise logical 'AND' and 'OR' operators as illustrated in Figure 3. The AND operation reinforces common connectivity structures while the OR operation retains any connectivity which exists between two randomly selected candidate nodes. Structural modifications occur less frequently than weight perturbations as suggested by Yao (1993) who advises that different time-scales should be applied at different levels of evolution. That is, variable connectivity should be considered a larger evolutionary step and occur less frequently than a smaller evolutionary step such as weight perturbations.

The output weights are found deterministically. The optimal (in a least-squares sense) output weight set is determined using the pseudoinverse. Iterative deterministic methods such as the LMS rule are also appropriate for determining the weights from the hidden units to the output units.

Parent Node 1	Operation	Parent Node 2		Offspring
[1111 0010]	AND	[1110 0110]	→	[1110 0010]
[1111 0010]	OR	[1110 0110]	→	[1111 0110]

Figure 3. *Bitwise logical operations applied to the candidate node connectivity strings.*

6 A Time-Series Modeling Example

Evolutionary learning was applied to the CCLA in an effort to model the sunspot time-series data set. This data set has served as a benchmark for a variety of statistical and neural network models. The average relative sunspot number represents a daily mean value taken from up to fifty observing stations throughout the world. The sunspot data set is typically broken down into a training set (years 1700-1920) and two test sets (years 1921-1955 and 1956-1979). This distinction results from the different statistical characteristics between the two test sets. The normalized MSE (NMSE) is used to evaluate the performance of the model which generates next step predictions. The NMSE is given by

$$NMSE = \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

where $\hat{y}$ is the single-step prediction and $\bar{y}$ is the mean of the target values. An *NMSE=1* implies that the estimate is just the average of the target values. For the sunspot data set, the NMSE is referenced as the MSE of each data segment scaled by the variance of the full data set

$$NMSE = \frac{1}{N \cdot \hat{\sigma}^2_{all}} \sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Pruning was not employed in the initial CCLA-EP experiment which consisted of 100 parents (candidate nodes) and 100 evolutionary training iterations. The experiment was arbitrarily stopped after the addition of three hidden units. We note that as additional hidden units are added, the performance of each model on a validation set could be used as a termination criterion. This idea is easily implemented since a new model is built on the existent one. Table 1 shows that the performance of the relatively more complex model (as demonstrated by the higher number of parameters) is comparable to results generated using regularization techniques in other investigations.

The next set of CCLA-EP experiments incorporated pruning during network construction and consisted of 50 parents with 100 evolutionary training cycles. Training was arbitrarily stopped after four hidden units were added. Again, a validation set could have been used as a stopping criterion. For the pruned network, good generalization occurs on both test sets with roughly half as many parameters as the unpruned CCLA-EP network as given in Table 1. Figure 4 shows the test and training results for the CCLA with complexity regularization.

Table 1. NMSE Performance and complexity of various sunspot models.

	NMSE Train: 1700-1920	NMSE Test: 1921-1955	NMSE Test: 1956-1979	No. of parameters
Tong & Lim (1980)	0.097	0.097	0.28	16
Weigend et al. (1992)	0.082	0.086	0.35	43
Svarer et al. (1993)	0.090	0.082	0.35	12-16
Deco et al. (1994)	0.091	0.087	0.32	n/a
CCLA-EP (3 hid. nodes, not pruned)	0.084	0.082	0.36	58
CCLA-EP (4 hid. nodes, pruned)	0.094	0.083	0.25	27*

** 25 if output units are pruned. See text.*

The linear (AR) portion of the model describes most of the dynamics of the sunspot series. The linear model is augmented by the nonlinear (sigmoidal) components of the CCLA as shown in Figure 5. The weights corresponding to the pruned architecture are given in Table 2. The 10 year difference between the *x(k-2)* and *x(k-12)* inputs to the first hidden unit corresponds with the cycle of maximum power spectral content for the first 280 years (1700-1979). More recent observations (1980-1987) of sunspot activity indicates that the maximum power spectral content corresponds to an 11 year cycle for the first 288 years (1700-1987).

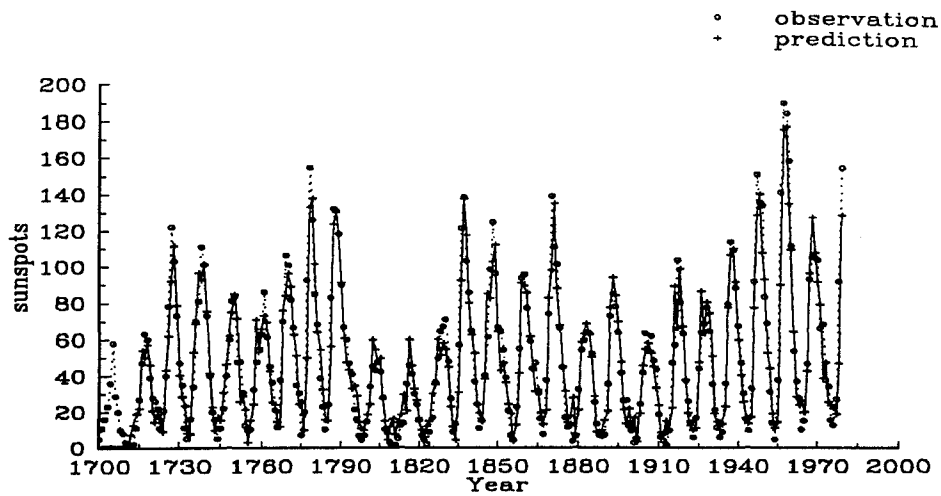
Additional pruning may take place on the output vector. For example, zeroing out the two smallest output weights listed in Table 2 yields the same performance values given in Table 1 and reduces the model complexity to 25 parameters.

7 Conclusion

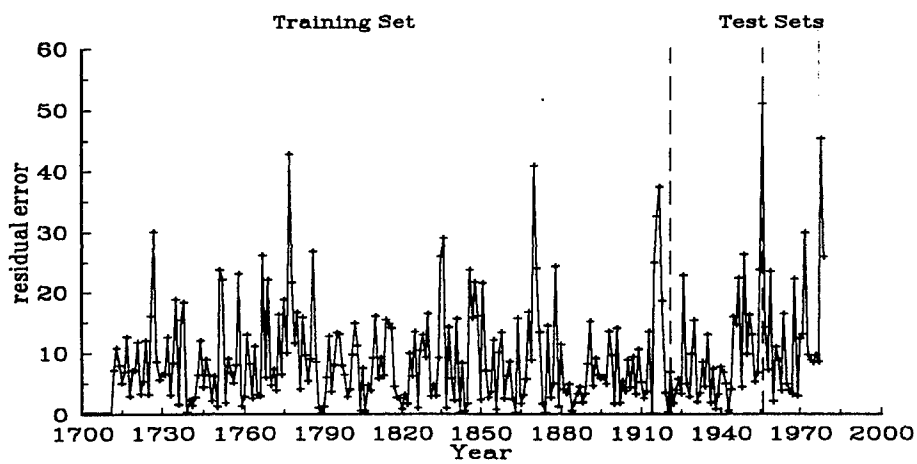
Pruned cascade-correlation learning architectures represent a combined linear+nonlinear time-series modeling technique with potentially better generalization capabilities. Evolutionary search is appropriate for simultaneously determining both hidden unit input structure and weights. The reader is cautioned not to draw any statistical conclusions for the presented example since Marple (1987) points out that time-series models "from short data records is a difficult problem in general." Additionally, the shift in the maximum power spectral energy content cycle observed for the example data set implies a time-invariance which is not addressed with the proposed modeling approach.

8 Acknowledgments

The authors gratefully acknowledge the support of Dr. Al Gordon, NRaD Deputy for Science. Special thanks to David Fogel, Tom English and the anonymous reviewers for their comments and suggestions.



(a)



(b)

Figure 4. *Results of the pruned, 4 hidden node CCLA with evolutionary learning. (a) An approximation of the sunspot data over the years 1712-1979 and (b) the magnitude of the residual error for each year.*

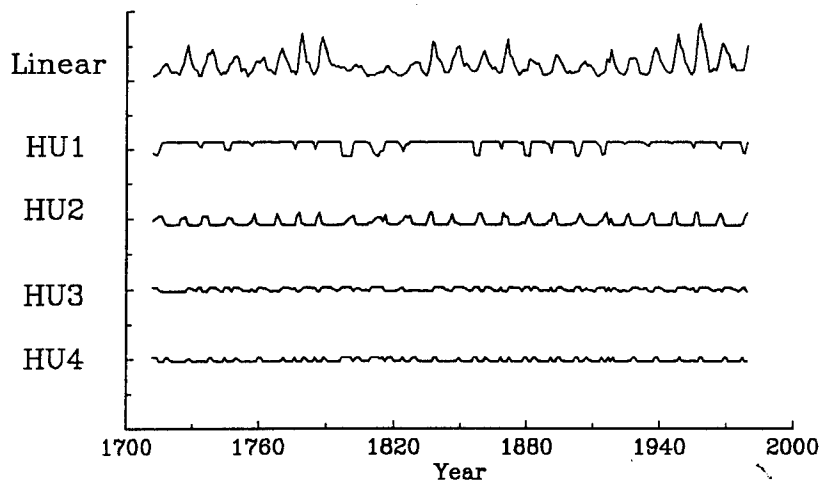


Figure 5. *The relative contributions of the linear portion and each nonlinear component of the pruned CCLA. The hidden units are designated by HU and the order in which they were incorporated.*

Table 2. The hidden unit and output weights for the pruned CCLA sunspot model.

HU Wts	*x(k-2)*	*x(k-8)*	*x(k-11)*	*x(k-12)*	HU1	HU2	HU3
HU1	11.318	-	-	-34.448	-	-	-
HU2	-	-	25.289	-13.869	-	-	-
HU3	12.231	-	-	-		8.719	-
HU4	-	22.981	-	-	-8.998	-11.681	-

Output Wts	bias	*x(k-1)*	*x(k-2)*	*x(k-3)*	*x(k-4)*	*x(k-5)*	*x(k-6)*	*x(k-7)*
	0.0524	-0.086	0.171	-0.062	0.129	-0.001	0.028	0.001
x(k-8)	*x(k-9)*	*x(k-10)*	*x(k-11)*	*x(k-12)*	HU1	HU2	HU3	HU4
-0.052	0.143	-0.159	-0.247	0.878	-0.103	-0.091	0.027	-0.029

9 References

Bäck, T., and Schwefel, H.-P. (1993). "An overview of evolutionary algorithms for parameter optimization," *Evolutionary Computation*, Vol. 1, No. 1, pp. 1-24.

Brooks, S.H. (1958), "A discussion of random methods for seeking maxima," *Operations Research*, 6, pp. 244-251.

Casdagli, M., and Eubank, S. (eds.) (1992), *Nonlinear Modeling and Forecasting*, Santa Fe Institute XII, Addison-Wesley.

Cichocki, A., and Unbehauen, R. (1993), *Neural Networks for Optimization and Signal Processing*, John Wiley & Sons.

Deco, G., Finnoff, W., and Zimmermann, H.G. (1994). "Unsupervised mutual information criterion for elimination of overtraining in supervised multilayer networks," submitted to *Neural Computation*.

Fahlman, S.E., and Lebiere, C.L. (1990). "The cascade-correlation learning architecture," Technical Report CMU-CS-90-100, Carnegie Mellon University, Pittsburgh, PA.

Fogel, D.B. (1991). *System Identification through Simulated Evolution: A Machine Learning Approach to Modeling*, Ginn Press.

Fogel, D.B. (1992). Evolving Artificial Intelligence, Ph.D. Dissertation, University of California, San Diego.

Fogel, L.J., Owens, A.J., and Walsh, M.J. (1966). *Artificial Intelligence through Simulated Evolution*, John Wiley & Sons.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*, MacMillan.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*, MIT Press.

Marple, S.L. (1987). *Digital Spectral Analysis with Applications*, Prentice-Hall.

Rissanen, J. (1986). "Stochastic complexity and modeling," *The Annals of Statistics*, Vol. 14, No. 3, pp. 1080-1100.

Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models*, John Wiley & Sons.

Svarer, C. , Hansen, L.K., and Larsen, J. (1993). "On design and evaluation of tapped-delay neural network architectures," *IEEE Int. Conf. on Neural Networks*, San Francisco.

Tong, H. and Lim, K.S. (1980). "Threshold autoregression, limit cycles and cyclical data," *Journal Royal Statistical Society B*, Vol. 42.

Weigend, A.S., Huberman, B.A., and Rumelhart, D.E. (1992). "Predicting sunspots and exchange rates," in M. Casdagli and S. Eubank (eds.), *Nonlinear Modeling and Forecasting*, Addison-Wesley.

Weigend, A.S., and Gershenfeld, N.A. (1993). "The future of time-series: learning and understanding," in A.S. Weigend and N.A. Gershenfeld (eds.), *Time-Series Prediction: Forecasting the Future and Understanding the Past*, pp. 1-70, Addison-Wesley.

Yao, X. (1993). "A review of evolutionary artificial neural networks," *International Journal of Intelligent Systems*, Vol. 8, No. 4, pp. 539-567.